AD-A236 821

ITATION PAGE

Form Approved OMB No. 0704-0188

ed to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, viewing the collection of information. Send comments regarding this burden estimate or any other aspect of this urden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson ffice of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

RT DATE: April 1991

3. REPORT TYPE AND DATES COVERED: technical

4. TITLE AND SUBTITLE

TORCS: A Teleoperated Robot Control System for the Self Mobile Space Manipulator

5. FUNDING NUMBERS

6. AUTHOR(S)

M. Ueno, W. Ross, and M. Friedman

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

8. PERFORMING ORGANIZATION REPORT NUMBER

CMU-RI-TR-91-07

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release;
Distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

This document describes the Teleoperated Robot Control System (TORCS) developed to control the Self Mobile Space Manipulator (SM) robot which is being developed in the Vision and Autonomous Systems Center. This robot is a semi-autonomous free-walking system being developed for space applications. TORCS provides a remote operator with comprehensive control of the robot on a number of different levels ranging from traditional teleoperation to completely autonomous walking. The heart of the TORCS system is an interactive, real-time X window based display tool which supports the controls for the robot and displays information on robot position and status. TORCS supports several well-integrated devices for teleoperation including a joystick, a six degree of freedom polhemus, and an isomorphic master controller. The operator can also perform teleoperation by specifying robot joint angles with the workstation mouse. Autonomous features include the ability to take single or multiple steps independently of operator intervention and to plan routes from one location to another. At the highest level, the operator need only specify a destination and TORCS will handle the rest. Provisions are included for the operator to oversee autonomous operations and to interrupt any operation in progress. The TORCS display includes a three-dimensional view of the robot during operation as well as close-up views of the robot gripper and readouts of exact robot status.

14. SUBJECT TERMS

15. NUMBER OF PAGES: 27 pp

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
unlimited	unlimited	unlimited	

TORCS: A Teleoperated Robot Control System for the Self Mobile Space Manipulator

Miyuki Ueno
William Ross
Mark Friedman

CMU-RI-TR-91-07

A-1

Vision and Autonomous System Center
The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

April 1991

This research was generously sponsored by Shimizu Corporation of Japan. The views and conclusion contained in this document are those of the authors and are not necessarily those of Shimizu Corporation or Carnegie Mellon University.

Contents

List of Figures

Abstract

This document describes the Teleoperated Robot Control System (TORCS) developed to control the Self Mobile Space Manipulator $(SM)^2$ robot which is being developed in the Vision and Autonomous Systems Center. This robot is a semi-autonomous free-walking system being developed for space applications. TORCS provides a remote operator with comprehensive control of the robot on a number of different levels ranging from traditional teleoperation to completely autonomous walking. The heart of the TORCS system is an interactive, real-time X windows based display tool which supports the controls for the robot and displays information on robot position and status. TORCS supports several well-integrated devices for teleoperation including a joystick, a six degree of freedom polhemus, and an isomorphic master controller. The operator can also perform teleoperation by specifying robot joint angles with the workstation mouse. Autonomous features include the ability to take single or multiple steps independently of operator intervention and to plan routes from one location to another. At the highest level, the operator need only specify a destination and TORCS will handle the rest. Provisions are included for the operator to oversee autonomous operations and to interrupt any operation in progress. The TORCS display includes a three-dimensional view of the robot during operation as well as close-up views of the robot gripper and readouts of exact robot status.

TORCS: A Teleoperated Robot Control System for the Self Mobile Space Manipulator

The Teleoperated Robot Control System, called "TORCS" for short, has been developed to operate the Self Mobile Space Manipulator ($(SM)^2$) which was developed at the Robotics Institute of Carnegie Mellon University. This document has been written for operators who control the $(SM)^2$ and consists of two parts: an overview of the Teleoperated Robot Control System and a users manual for the screen interface. Part One, the system overview, describes the architecture of the robot interface and gives an introduction to the window-based control program. The second part gives detailed instruction in the usage of the TORCS interface.

Part One -- System Overview

1.1 The Self Mobile Space Manipulator $(SM)^2$

Robots on the planned NASA space station have the potential to assist astronauts during Extra-Vehicular Activity (EVA), and to replace astronauts in performing simple, dangerous, and routine tasks. Appropriate tasks include transport of small parts, holding tools or lights, and carrying cameras or other sensors. To demonstrate the pertinent technologies, the Space Robotics Laboratory at the Robotics Institute of Carnegie Mellon University, is developing a robot for locomotion and basic manipulation on the planned space station. The robot comprises two flexible links connected by a rotary joint, with two degree-of-freedom wrist joints and grippers at each end. The grippers screw into threaded holes in the nodes of the space station trusswork, enabling the robot to walk by alternately shifting its base of support from one foot to the other in and end-for-end manner. In addition to the robot, we have developed a testbed including a 1/3 scale (1.67-meter modules) trusswork and a gravity compensation system to simulate a zero-gravity environment.

1.2 $(SM)^2$ Control Strategy

The Self Mobile Space Manipulator$(SM)^2$ has been designed to be a telerobotic system, combining features of both human guidance and autonomous operation. Our control architecture is hierarchical and provides for control on a number of different levels, from the low-level, which is joint-specific teleoperation by an operator, to the higher-levels, such as semi-autonomous operation. Several human-machine interfaces, such as a joystick and a five degree of freedom hand controller, are available to the teleoperator. TORCS has been developed to provide the human operator the ability to choose the level of control, for a specific task, with which he/she feels most comfortable.

1.3 Structure of TORCS

As shown in figure 1.1, TORCS consists of four parts: the **Screen Interface**, the **Command Module**, the **Robot Interface** (which is connected to the Real-Time Controller for the $(SM)^2$), and the **Robot Simulator**. TORCS implements the entire user-interface for the robot while actual control of the robot is handled by real-time software which is not part of TORCS. Communications with the Real-Time Controller, running under the CHIMERA operating system, is accomplished through a UNIX socket-based protocol. The CHI-

MERA system is typically run on an IRONICS 68020 based CPU board hosted on a Sun 3 machine while the TORCS system runs on a color Sparcstation. Because of the socket-based communication, TORCS may run on virtually any X11 platform on the same network as the CHIMERA host.

1.3.1 Screen Interface

The Screen Interface for TORCS is an X11 based tool which communication between users and the $(SM)^2$. The operator uses this interface, in conjunction with joysticks and other input devices, to effect control of the robot. Additionally, this interface presents the robot status and the position information on several graphical displays. A detailed description of this interface, and its use, is presented in the second part of this document.

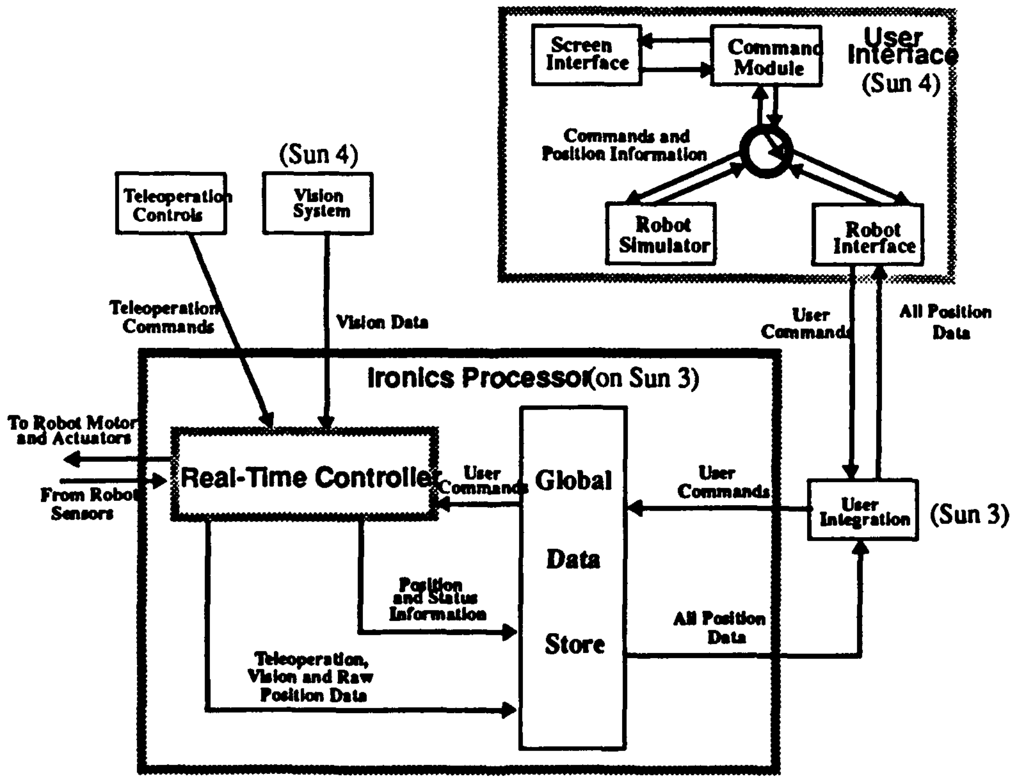
1.3.2 Command Module

The Command Module provides higher-level control for the robot such as path-planning, supervision of multiple-step movements and other functions which are composed of multiple calls to the Real-Time Controller. As the Command Module sits between the Screen Interface and the robot, it also interprets user commands and dispatches them to the Robot Interface or the Robot Simulator.

1.3.3 Robot Interface

The Robot Interface section of TORCS handles communication with the Real-Time Controller. All data passed between the controller and the Command Module is handled and buffered by this module. However, the Robot Interface module does not communicate directly with the Real-Time Controller. Communication is performed through a UNIX daemon (labeled "User Integration" in figure 1.1) which allows the TORCS

Figure 1.1 Software Architecture for $(SM)^2$

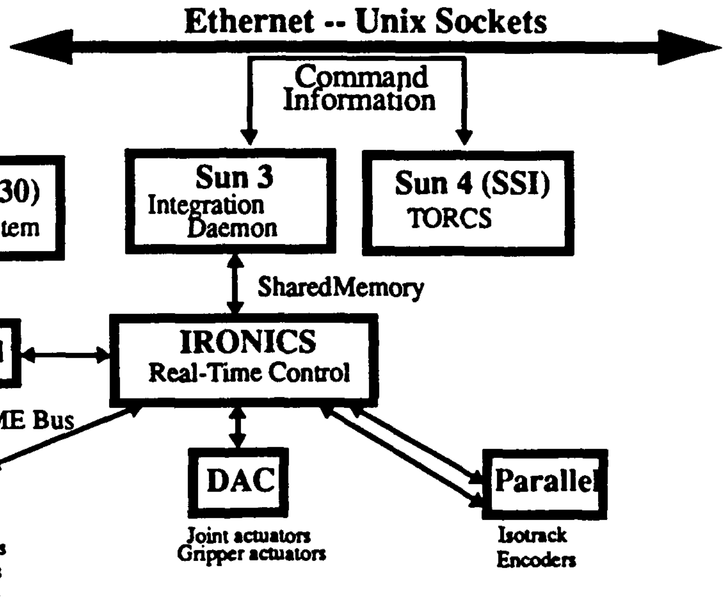


software to run on a different workstation than the one which hosts the CHIMERA based the Real-Time Controller. The UNIX daemon runs on the CHIMERA host and communicates with TORCS via UNIX tcp/ip sockets.

The communication between the Command Module and the Real-Time Controller operates as follows: When the Command Module wants to fetch data from the Real-Time Controller, the User Integration daemon is called by the Robot Interface module. When the daemon gets a request from the Robot Interface, it collects the required data from the Global Data Store (the Real-Time Controller updates the sensory information in the Global Data Store 10 times per second) and then transmits the data to the Robot Interface.

When the Command Module has a command for the Real-Time Controller, it is sent to the daemon through the Robot Interface. When this command is received, it is written into the Global Data Store and a flag is set to tell the Real-Time Controller a command is waiting. The Real-Time Controller regularly checks the status of this flag and acts on the commands received.

Figure 1.2 Hardware Configuration for $(SM)^2$



1.3.4 Robot Simulator

The current version of TORCS does not provide a Robot Simulator module. The Robot Simulator is intended to be a separate program which provides kinematic and dynamic simulation of the $(SM)^2$ system, thus allowing development of the control software to proceed without the necessity of actually operating the robot system. Communications between TORCS and the simulator would be performed in the same manner as that with the Real-Time Controller -- allowing TORCS to operate without regard as to whether the robot is real or simulated.

1.4 Structure of the Screen Interface

The Screen Interface is divided into two parts: the upper half of the window, called the "Command Panel", and the lower half of the window, called the "Monitor Panel" (see figure 1.3). On the **Command Panel** are many buttons which are used to specify parameters for walking, for simple teleoperation and to compose commands to be sent to the Real-Time Controller or to the Robot Simulator. The purpose of the **Monitor Panel** is to display the current state of the robot to an operator. The current state of the robot is updated five times a second in the Over-View and Fine-View Windows. The Over-View Window is also used to specify the target state of the robot.

1.4.1 The Command Panel

The Command Panel is divided into four parts: the **Mode Panel**, the **System Panel**, the **Parameter Panel** and the **Action Panel.**

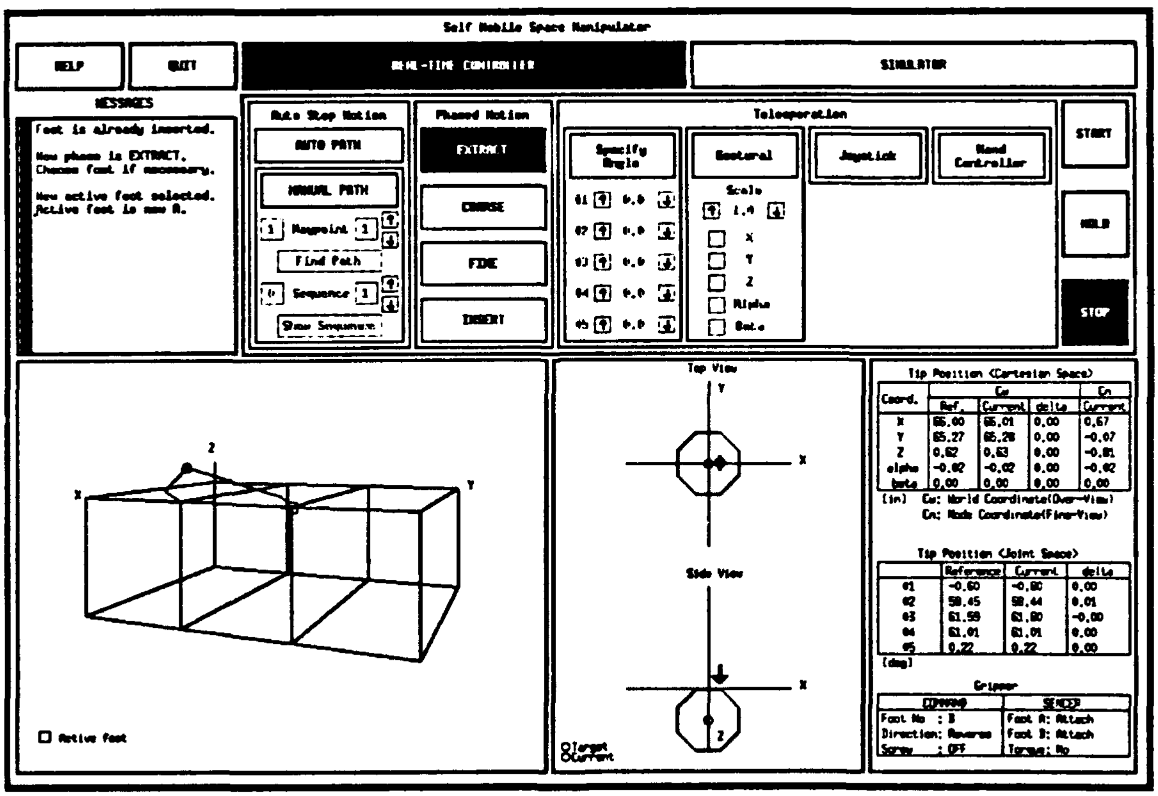
1) The Mode Panel

The two buttons at the top of the Screen Interface are used to select whether the system is to communicate with the robot or with a robot simulator. The default is set to "REAL-TIME CONTROLLER". This means that all parameters an operator specifies are commands to the actual robot hardware. When the operator wants to use the Robot Simulator, he/she does the following:

1. Locate the arrow of the mouse (or the track ball) on the "SIMULATOR" button.
2. Click the left button of the mouse (or the track ball).

Note: In the current version, the Robot Simulator is not available.

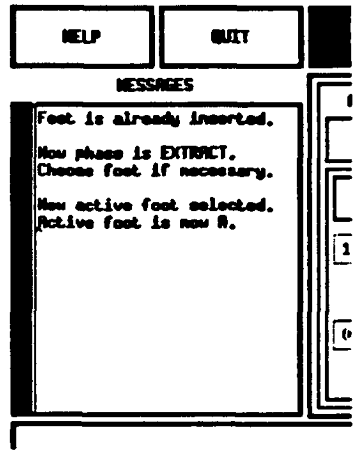
Figure 1.3 Screen Interface of the Teleoperated Robot Control System



2) The System Panel

This section of the system panel consists of two buttons and one window: the "QUIT" button, the "HELP" button and the "MESSAGES" window. These three features are described below.

Figure 1.4 Screen Interface System Panel



a) The "QUIT" Button
When an operator chooses this button, TORCS records the current state of the robot in the ASCII file named "SAVEDATA". This state consists of four parameters: the strut number on which the robot is sitting, the node number in which the base foot of the robot is inserted, the robot's orientation and the orientation of the base foot. This file is read when TORCS next runs to get the robot state. See section 2.1.1 for a detailed description of these parameters. After saving state, TORCS closes its window and exits.

Note: If the robot is moved before TORCS is restarted, "SAVEDATA" may contain incorrect data.

b) The "HELP" Button
The "HELP" button will allow the operator to get information about various parts of the Screen Interface. To get information on a particular button, first click the "HELP" button and then click on the button for which you need help. A short help message will be printed in the "MESSAGES" window.

c) The "MESSAGES" Window
All messages regarding the operation of the robot are displayed in this window. Look here for error messages, warnings or notes telling you what to do next.

3) The Parameter Panel

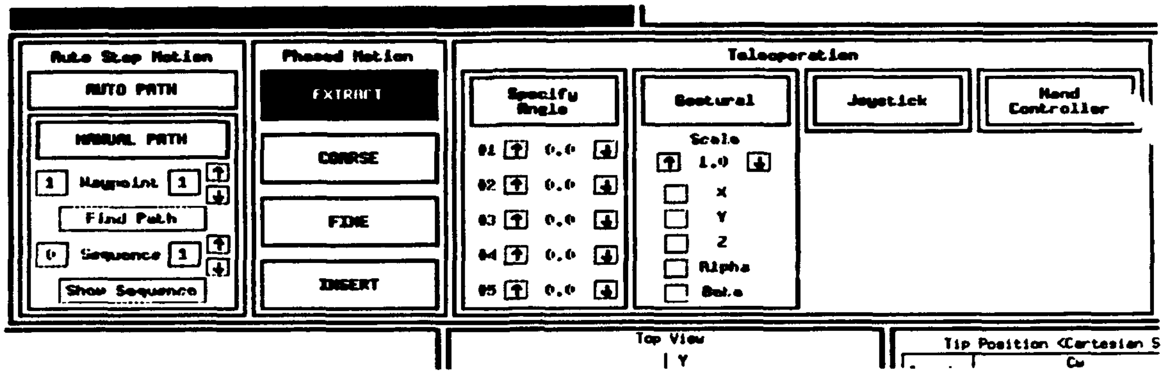
This panel is used to specify the parameters for all operations. The robot may be operated using any of three control paradigms: **Fully-Autonomous Operation**, **Semi-Autonomous Operation** and **Teleoperation**. The details of each follow.

a) Fully-Autonomous Operation (Auto Step Motion
In Auto Step mode, the destination of the robot is entered by the operator and the Command Module then guides the robot through a step or series of steps to reach that goal. The path to follow is generated automatically and movement proceeds without operator intervention.

b) Semi-Autonomous Operation (Phased Motion)
A robot step is composed of four phases: foot extraction, coarse motion, fine motion and foot insertion. In Phased Motion mode, each phase is controlled by the operator. When a phase is selected, the Command Module and the Real-Time Controller control the robot as necessary to complete the operation. This operation mode is useful for error recovery, demonstrations, testing and other special-case situations.

c) Teleoperation
Four methods of low-level robot control are provided for use by the operator. These methods all fall under the teleoperation control paradigm. Joint angles may be directly entered by the operator, a six degree of freedom polhemus may be used for gestural control of the robot, a joystick may be used, or, a special five degree of freedom hand controller may be used. In some case, several methods of control may be combined.

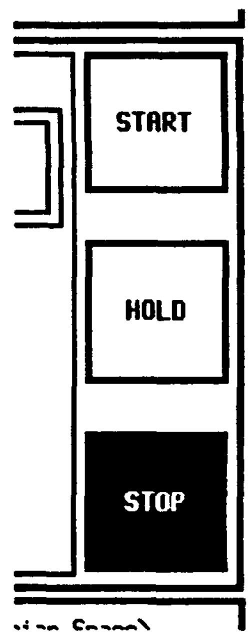
Figure 1.5 Parameter Panel



4) The Action Panel

This panel consists of two control buttons labeled "START" and "STOP", and one indicator labeled "HOLD".

Figure 1.6 Action Panel



a) The "START" Button
When the operator has specified a desired action, clicking on the "START" button will cause that action to commence. The button stays illuminated as long as the operation continues. If no action has been specified, the "START" button does nothing.

b) The "STOP" Button
This button indicates when an operation is complete. Clicking the button during an operation stops the operation in mid-stride. An auxiliary foot switch connected to the Real-Time Controller is also available for faster emergency stops. As this foot switch is monitored by TORCS, pressing it will also toggle the "STOP" button.

c) The "HOLD" Light
This indicator shows the status of the "HOLD" foot pedal. This pedal is used to pause an operation. Pressing this pedal causes the robot to freeze in position. Releasing the pedal allows the robot to continue the interrupted operation. The "HOLD" light does not function as a button but only displays the status of the pedal.

1.4.2 Monitor Panel

The **Monitor Panel** is divided into three parts: the **Over-View Window**, the **Fine-View Window** and the **Data Indicator**. These three displays are described below and use the following coordinate systems:

1) Coordinate Definitions

The several conventional coordinate systems used in the $(SM)^2$ system are defined as follows:

•World (Truss) Coordinates

World coordinates are relative to the trusswork and are the coordinates used in the Over-View Window.

•Robot Coordinates

This coordinate system is relative to the base of the robot. In the initial robot configuration (each joint angle equal to 0 degrees), the robot coordinates are as shown in figure 1.7 (a).

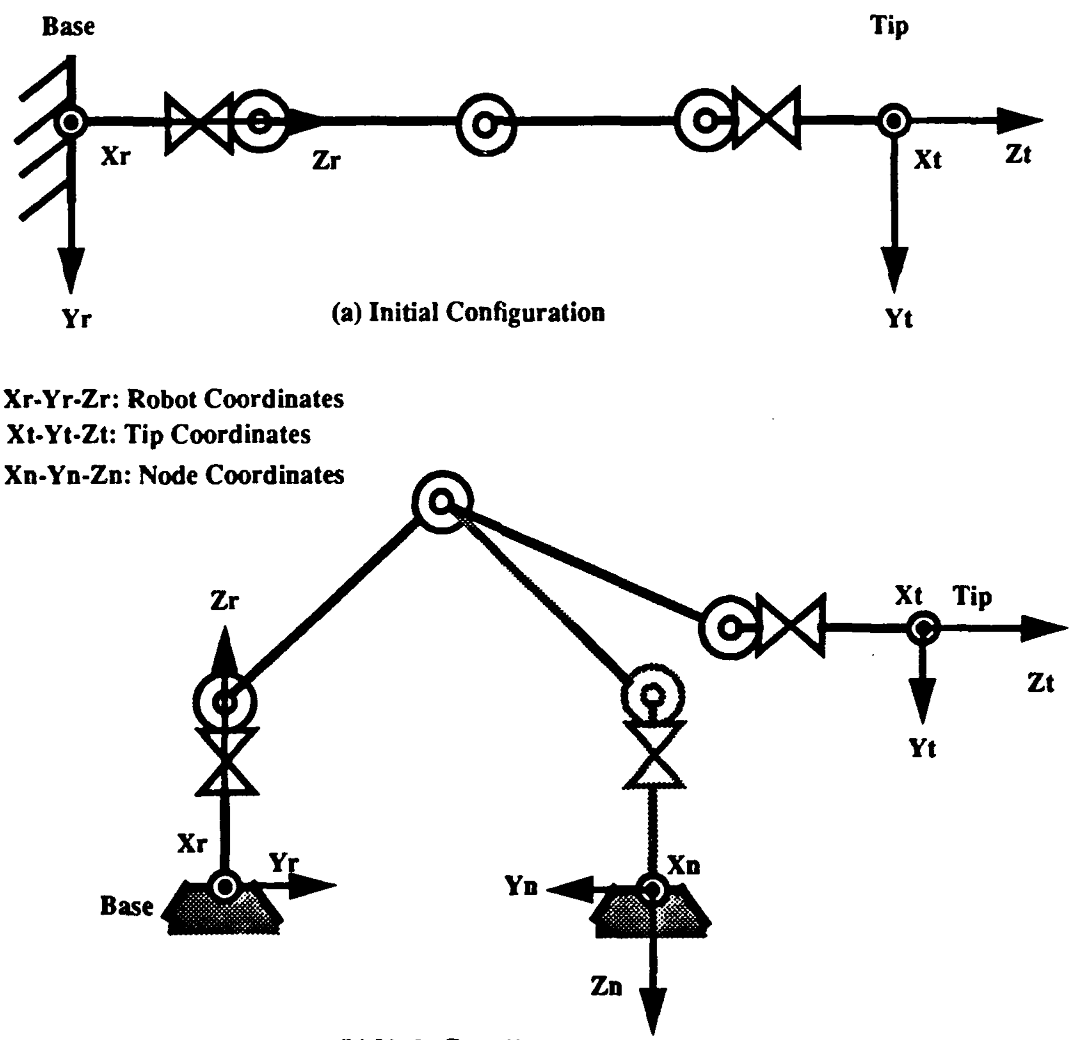
•Tip Coordinates

The Tip Coordinate system is relative to the tip of the robot (figure 1.7) and changes according to the pose of the robot. With the robot in the initial configuration, this coordinate system is equivalent to the Robot Coordinate system.

• Node Coordinates

This coordinate system is relative to the face of the node into which the tip of the robot is inserted.

Figure 1.7 Coordinate Definitions



(a) Initial Configuration

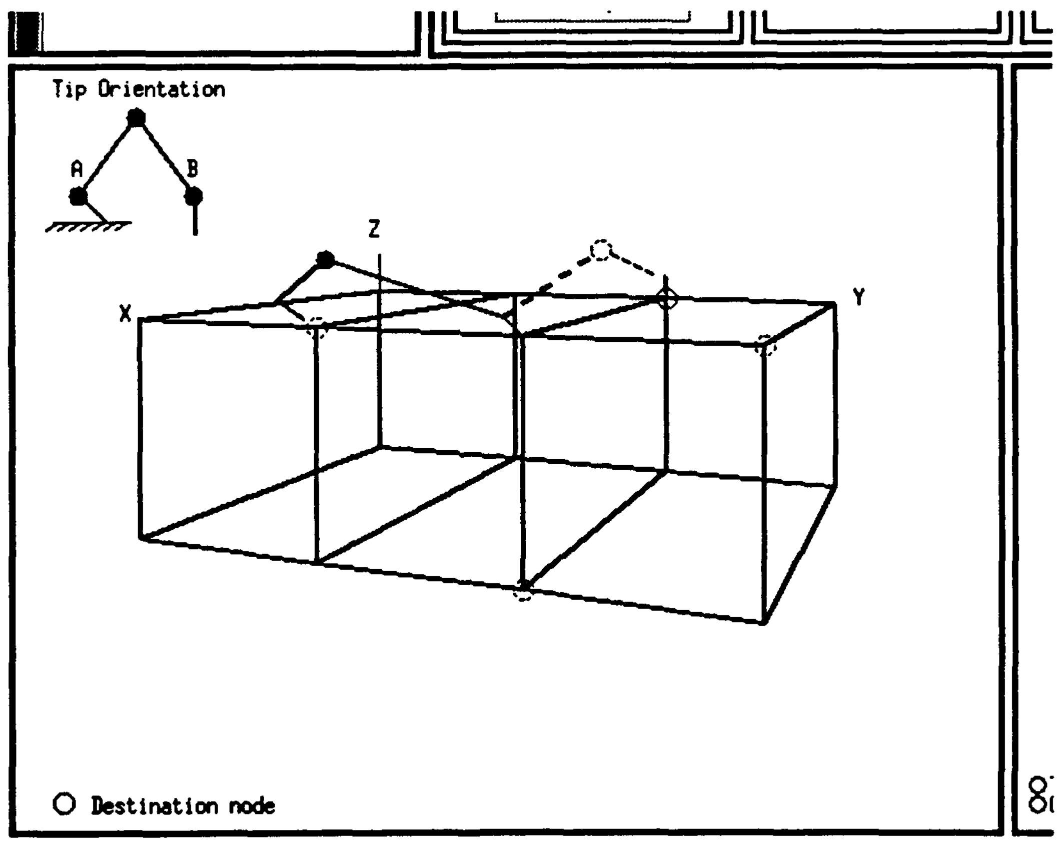
(b) Node Coordinates

2) The Over-View Window

This window shows a three dimensional view of the $(SM)^2$ on the Space Station trusswork. The robot is shown in its current position as a stick figure on the trusswork (on a color monitor the robot is violet and the trusswork is yellow). This window is also used by the operator to indicate destinations for the robot; this is covered in detail in section 2.1.1. The Over-View display is updated five times per second. The position information comes from the Real-Time Controller or the Robot Simulator as robot (joint) coordinates which are translated to Cartesian space data in world (truss) coordinates.

Note: The Real-Time Controller and the Robot Simulator both store and manipulate positions in robot coordinates. The Real-Time Controller and the Robot Simulator have no way of knowing the position of the robot with respect to the entire trusswork. This is why the "SAVEDATA" file must be used to specify an initial robot position on the truss.

Figure 1.8 The Over-View Window



3) The Fine-View Window

This window gives a close-up view of the target node and the position of the tip of the robot. The view is shown as a pair of two-dimensional representations: one in XY and the other in XZ. The robot tip position, relative to the node, is derived from vision system data. Also shown are the orientations of the tip twist and flex joints. The target position is displayed as crosshairs in both top and side views. If you are using a color display, the current position will be in light green and the target position will be violet.

Note: When the tip approaches a node (to within 5 inches), the Command Module displays that node in the Fine View window. Target position crosshairs are only shown when the node is the target node.

4) The Data Indicator

The Over-View Window and the Fine-View Window give an operator the approximate robot position. The precise position is shown in the Data Indicator window. This window shows both the current and the target tip positions and the difference between these positions in Cartesian space. Other information displayed in the window includes foot status for both feet and the status of the motors for grippers.

Figure 1.9 The Fine-View Window

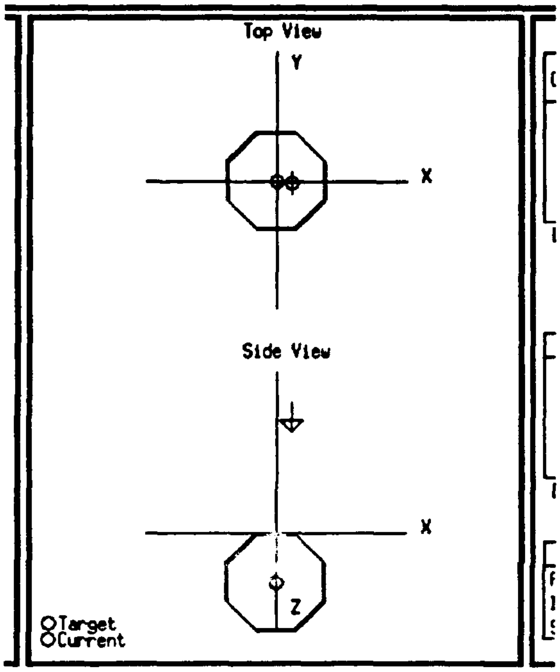
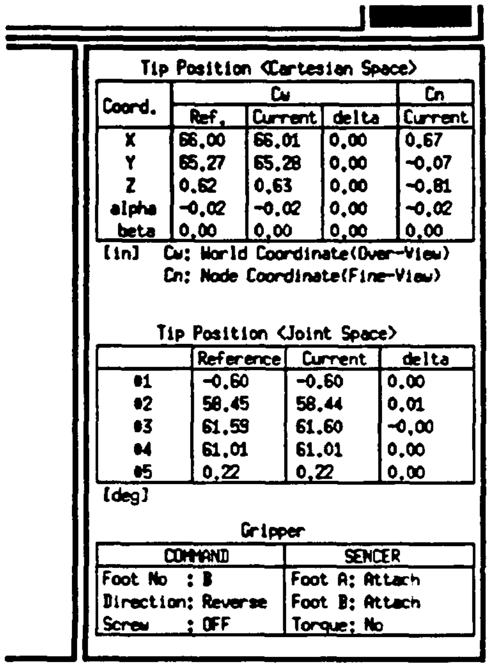


Figure 1.10 The Data Indicator



Tip Position <Cartesian Space>

Coord.	Cw			Cn
	Ref.	Current	delta	Current
X	66.00	66.01	0.00	0.67
Y	65.27	65.28	0.00	-0.07
Z	0.62	0.63	0.00	-0.81
alpha	-0.02	-0.02	0.00	-0.02
beta	0.00	0.00	0.00	0.00

[in] Cw: World Coordinate(Over-View)
Cn: Node Coordinate(Fine-View)

Tip Position <Joint Space>

	Reference	Current	delta
#1	-0.60	-0.60	0.00
#2	58.45	58.44	0.01
#3	61.59	61.60	-0.00
#4	61.01	61.01	0.00
#5	0.22	0.22	0.00

[deg]

Gripper

COMMAND	SENCER
Foot No : B	Foot A: Attach
Direction: Reverse	Foot B: Attach
Screw : OFF	Torque: No

Part Two -- Using The Screen Interface

This chapter details the use of the TORCS screen interface. In the current version of TORCS, no robot simulator has been provided. If "SIMULATOR" is chosen at the top of the TORCS window, the system remains connected to the actual robot.

2.1 Autonomous Motion

Before describing the operation of the Autonomous Motion features, this section introduces the basic concepts needed for the operation of the robot.

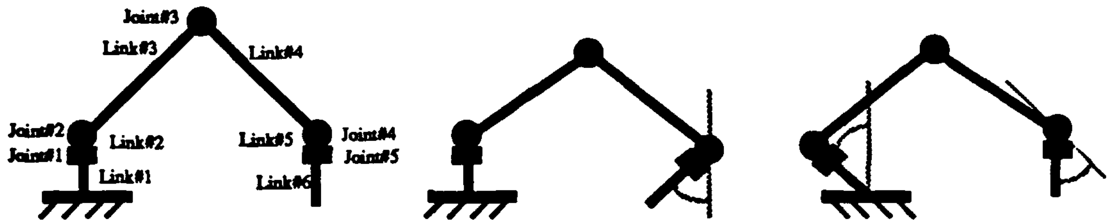
2.1.1 Parameters to describe the state of $(SM)^2$

When both ends of the $(SM)^2$ are inserted into nodes, the $(SM)^2$ can adopt three different configurations as shown in figure 2.1. These configurations are defined by the angles of joints 2, 3 and 4. The angles of joints 1 and 5 do not effect the configuration. Considering these configurations, we made a step catalog (see Appendix B). A single step is a movement from one configuration to the same or a different configuration. Using only these three configurations, the $(SM)^2$ can reach any strut of the trusswork and can adopt any orientation to the trusswork. Currently, however, we only allow walking outside of the trusswork. The TORCS system should thus allow the operator to specify any target configuration and should be able to derive the sequence of steps necessary to reach it. The configuration of the $(SM)^2$ is described by two parameters: the base orientation and the tip orientation (see figure 2.1). Also used to specify robot position are parameters which describe what type of step will be taken and the kind of kinematic translation function between the world coordinate and the robot coordinate to be used. The details for these parameters are as follows:

- **Base Orientation:** This is the angle between the strut on which the $(SM)^2$ is sitting and link 1 (see figure 2.1). Only two orientations 0 degrees and 45 degrees are allowed currently.
- **Tip Orientation:** The describes the angle between link 1 and link 6. If the base orientation is 0 degrees, two types of orientation, 0 degrees and -45 degrees, are allowed as the tip orientation. If the base orientation is 45 degrees, the tip orientation can only be -45 degrees.
- **Strut Number:** Each strut on the trusswork has a unique identification number. The strut number of the robot indicates the strut which connects the two nodes to which the robot is attached.
- **Robot Plane:** This describes the orientation of the robot with respect to world (truss) coordinates. We define a plane made by the current strut and the $(SM)^2$ as the "robot plane". The vector perpendicular to the strut and intersecting with the knee joint of the robot is used to describe the orientation of the robot plane. The various robot planes are defined in figure 2.2.
- **Base Node Number:** This is the identifying number of the node into which the current base foot of the robot is inserted.

These five parameters completely describe the state of the $(SM)^2$.

Figure 2.1 The Configuration of the $(SM)^2$

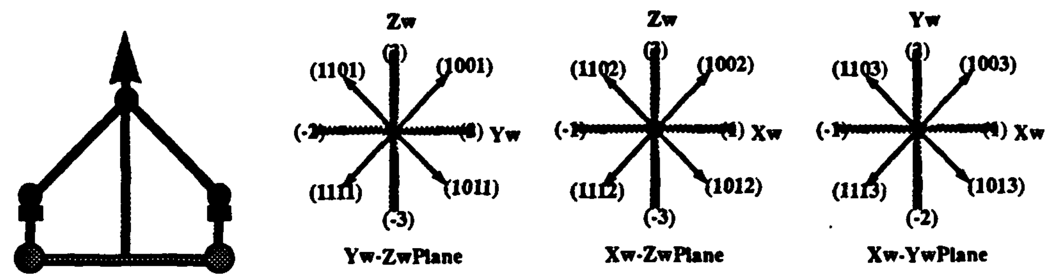


(a) Base: 0[deg]
Tip: 0[deg]

(b) Base: 0[deg]
Tip: -45[deg]

(c) Base: 45[deg]
Tip: -45[deg]

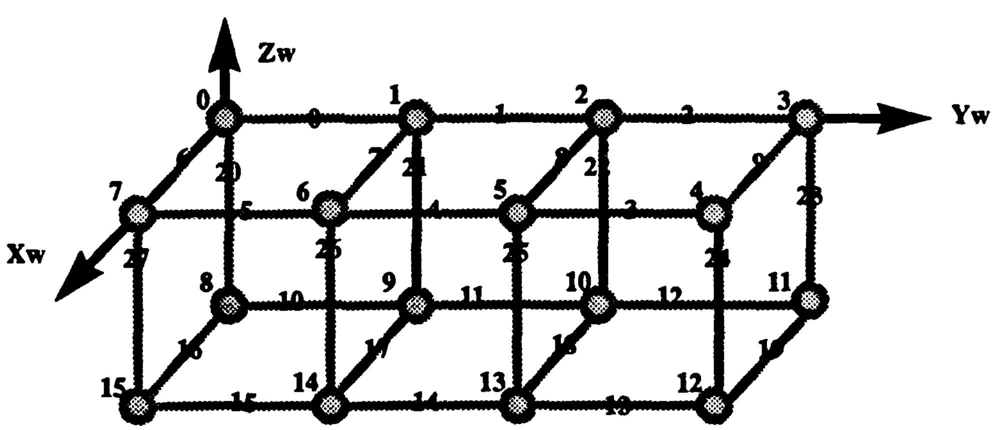
Figure 2.2 Robot Plane Number Definition



(a) Robot Plane Vector

(b) Relation between Robot Plane Number and Robot Plane Vector

Figure 2.3 Numbering of Struts and Nodes



2.1.2 The Strut Sequence Generator

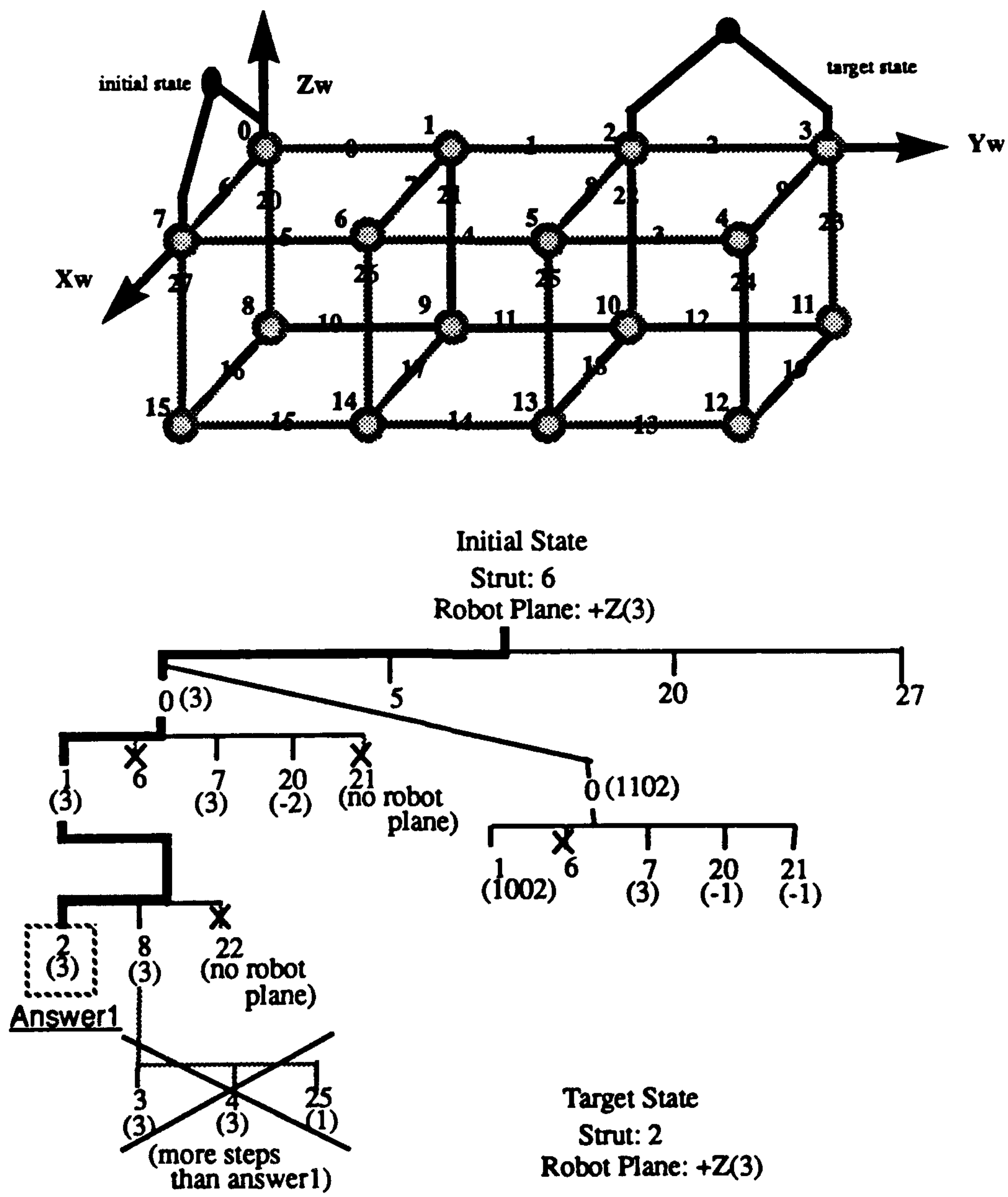
When an operator specifies a target state of the robot, a path must be chosen from the current state to the target state. Since the trusswork is completely symmetrical, there often exist many paths to a given target. TORCS uses a path generator to find and display each of the short paths to the target. A strut sequence is a list of strut numbers that defines a path for the walking on the trusswork. We call the path generator the

"Strut Sequence Generator". While TORCS will allow the operator to specify any path to be taken by the robot, coarse level movements within a step cannot be specified as all movement phases are controlled by the Real-Time Controller.

1) Strut Database

At start-up, TORCS loads the file "STRUT_DBASE" which contains a description of the trusswork. In this file, each node is defined by its number. The struts on the trusswork are defined by the pairs of nodes which they connect. Appendix C gives more information of the format of this file.

Figure 2.4 Searching Tree (example)



2) The Path Generation Algorithm

A path on the trusswork can be described as a tree structure as shown in figure 2.4. A "depth-first" searching technique is used in the Strut Sequence Generator to search the path tree for paths to the target node. The searching procedure is as follows:

Each path down the tree is followed until a state matching the target state is found. For each such path, the number of steps in the path and the time to travel the path are calculated (different types of steps take different amounts of time to complete). Tree branches resulting in longer paths than those already found are ignored as are paths which result in loops (passing the same strut twice). The generator stops its search when all branches have been explored or rejected.

The Strut Sequence Generator works on any truss configuration.

2.1.3 "AUTO PATH" Mode and "MANUAL PATH" Mode

Autonomous Step Motion is performed in one of two modes: "AUTO PATH" mode or "MANUAL PATH" mode. "AUTO PATH" mode is the default when TORCS is initialized. In "AUTO PATH" mode, TORCS automatically selects a path to the target position. In this case, only the target configuration needs to be specified. Before committing the robot to start the operation, the chosen path can be viewed by clicking the "Show Sequence" button. In "MANUAL PATH" mode, the operator may set up to 9 waypoints for the robot to pass through. Additionally, the operator may select from a number of automatically generated paths. Further details on this operation can be found in section 2.1.4.

2.1.4 Instructions for Setting Robot Parameters

Five parameters are needed to describe the state of the $(SM)^2$. These are 1) the base orientation, 2) the tip orientation, 3) the strut number, 4) the robot plane and 5) the base node number. These parameters are set using the left mouse (or track-ball) button as described below:

Setting Parameters to Describe the Target State of the $(SM)^2$ in Auto Step Mode

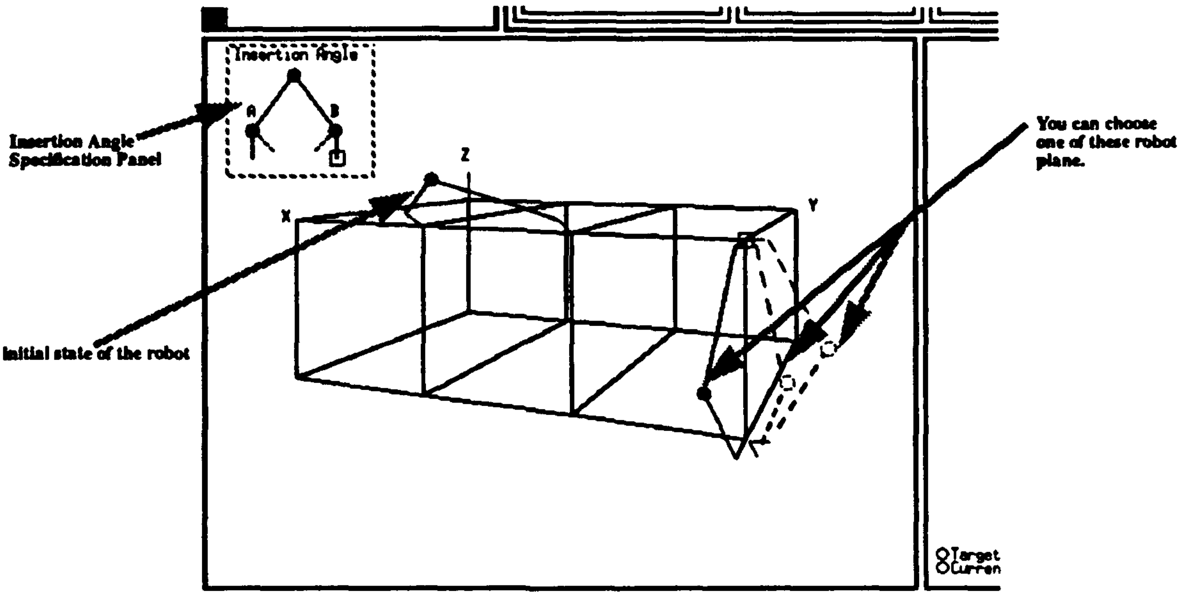
1. **In the Over-View Window, click on two nodes to specify the target strut.**
   The color of the strut you choose will turn light green.
   Around the strut you specify, several light purple robots will appear. These robots indicate the various robot planes possible for that strut.
   At the same time the Insertion Angle Specification Panel will appear in the upper left corner (See figure 2.5).

2. **Select the knee joint of one of the robots which appear around the strut to specify the target robot plane.** The
   Insertion Angle Specification Panel will be redrawn to reflect the robot plane specified. A default robot plane is shown as a light purple robot drawn with solid lines. If you want the default, no action is necessary.

3. **Use the Insertion Angle Specification Panel to set the target insertion angle of each foot.**
   The targeted angle is indicated by a small square on the foot. As with the target robot plane, a default angle is shown for each foot by solid lines.

   Note: Since the Screen Interface automatically determines which node should be the base for each step, the operator does not set the base node number for the target configuration.

Figure 2.5 Setting the Parameters for Auto Step Motion



2.1.5 Using Auto Step Motion

The controls for Auto Step Motion are found on the left hand side of the Parameter Panel. Instructions for using the two Auto Step modes are given below.

1) "AUTO PATH" Mode

1. **Select the "AUTO PATH" button.**
   When TORCS is started, the "AUTO PATH" button on the Parameter Panel is selected as the default. If both feet of the robot are not inserted, an error message will appear in the "MESSAGES" Window.

2. **Set the parameters to describe the target state of the robot (as described in section 2.1.4).**
   Once the operator has set a target strut, the "START" button on the Action Panel is enabled.

3. **To review the path from the current state to the target state, click the "Show Sequence" button.**
   The planned path will be shown in the Over-View Window. If you have made an error in setting the target state, a message will appear in the "MESSAGES" Window. In this case you need to carefully reset the parameters.

4. **If you are ready, click the "START" button on the Action Panel to make the $(SM)^2$ walk.**
   The "HOLD" light should be on, indicating that the robot is ready but holding. (Never step on the "HOLD" foot switch before clicking the "START" button.)

5. **Step on the "HOLD" foot switch to enable the robot.**

6. **If you want to stop the robot or when the walk is completed, release the "HOLD" foot switch first and then click the "STOP" button on the Action Panel.**

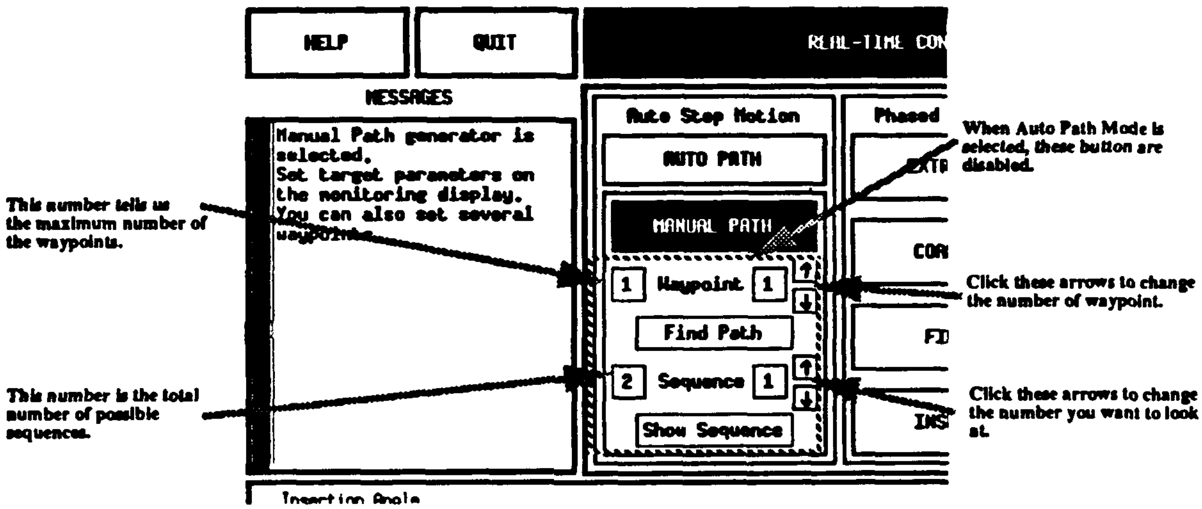
The two foot switches, "STOP" and "HOLD", are used in every operation. The "HOLD" switch must always be down to allow the robot to continue with the current operation. Pressing the "STOP" switch at any time acts to cancel the current operation.

2) MANUAL PATH Mode

1. Select the "MANUAL PATH" button.

Selecting the "MANUAL PATH" button enables several additional controls in the Auto Step Motion portion of the display (See figure 2.6).

Figure 2.6 MANUAL PATH Mode



2. Set the parameters for each waypoint.

The window located on the left side of the letters "Waypoint" shows the maximum number of waypoints. The one on the right side shows the number of the waypoint for which you are currently setting parameters. You can increase/decrease this number by clicking the up or down arrow mark beside this window. Each waypoint is specified in the same way as target nodes specified while in "AUTO PATH" mode.

3. Find possible paths.

When you finish specifying the parameters to describe the state of the robot for each waypoint, click the "Find Path" button. This button runs the Strut Sequence Generator. A few seconds later, the number to the left of "Sequence" indicates the number of paths found.

4. To view sequences, click the "Show Sequence" button.

The number to the right of "Sequence" indicates the path being displayed. Pick a path to follow by clicking the up and down arrows to get to a path you like.

5. If you are ready, click the "START" button on the Action Panel to make the $(SM)^2$ walk.

The "HOLD" light should be on, indicating that the robot is ready but holding. (Never step on the "HOLD" foot switch before clicking the "START" button.)

6. Step on the "HOLD" foot switch to enable the robot.

7. If you want to stop the robot or when the walk is completed, release the "HOLD" foot switch first and then click the "STOP" button on the Action Panel.

2.2 Phased Autonomous Motion

Each step of the robot is comprised of four phases which, in Phased Autonomous Motion mode, can be performed independently. This gives the operator a method of control lying between autonomous operation and teleoperation. This capability can be useful under special circumstances or during emergencies.

2.2.1 Foot Extraction

This operation allows the operator to simply detach one of the feet of the robot from the connected node. Operation is as follows:

1. **Click the "EXTRACT" button on the Parameter Panel.**
   A square mark will appear on one of the robot feet in the Over-View Window indicating the foot to be extracted.
   If this operation is attempted while both feet are not inserted, an error message will appear in the "MESSAGES" window.

2. **Change the active foot, if necessary.**
   If you wanted to extract the other foot, simply click on it. The square indicator will move to this new foot.

3. **When ready, click the "START" button on the Action Panel to start the extraction.**
   The "HOLD" light should be on, indicating that the robot is ready but holding. (Never step on the "HOLD" foot switch before clicking the "START" button.)

4. **Step on the "HOLD" foot switch to enable the robot.**

5. **If you want to stop the robot or when the extraction is complete, *release the "HOLD" foot switch* first and then click the "STOP" button on the Action Panel.**

2.2.2 Coarse Motion

This operation allows the operator to move one of the robot's feet to the vicinity of a neighboring node. The foot must first be extracted if it hasn't already.

1. **Click the "COARSE" button on the Parameter Panel.**
   Dashed circles will appear around reachable adjacent nodes with a default destination node marked by a solid circle. A robot will also appear, showing the default target tip orientation.

2. **Choose the destination node, if necessary.**
   Clicking on a marked node will make it the destination node and redraw the display.

3. **Set the tip orientation, if necessary.**
   Use the Tip Orientation Panel to choose a different target tip orientation.

4. **When ready, click the "START" button on the Action Panel to start the movement.**
   The "HOLD" light should be on, indicating that the robot is ready but holding. (Never step on the "HOLD" foot switch before clicking the "START" button.)

5. ***Step on the "HOLD" foot switch to enable the robot.***

6. **If you want to stop the robot or when the movement is complete, release the "HOLD" foot switch first and then click the "STOP" button on the Action Panel.**

2.2.3 Fine Motion

Fine motion is the part of a step which lines the foot up for insertion. Fine motion is only used when the foot is in close proximity to the target node.

1. **Click the "FINE" button on the Parameter Panel.**
   When you click the "FINE" button, a circle mark will appear around the adjacent node in the Over-View Window. The $(SM)^2$ will approach this node using the vision system.
   If both ends are already inserted, an error message will appear in the "MESSAGES" window.

2. **When ready, click the "START" button on the Action Panel to start the movement.**
   The "HOLD" light should be on, indicating that the robot is ready but holding. (Never step on the "HOLD" foot switch before clicking the "START" button.)

3. **Step on the "HOLD" foot switch to enable the robot.**

4. **If you want to stop the robot or when the movement is complete, release the "HOLD" foot switch first and then click the "STOP" button on the Action Panel.**

2.2.4 Foot Insertion

When a foot is close enough to a node to be inserted, the foot insertion phase can be used to perform an insertion.

1. **Click the "INSERT" button on the Parameter Panel.**
   When you click the "INSERT" button, a circle mark will appear around the adjacent node in the Over-View Window. The tip will be inserted into this node.
   If both ends are already inserted, an error message will appear in the "MESSAGES" window.

2. **When ready, click the "START" button on the Action Panel to start the insertion.**
   The "HOLD" light should be on, indicating that the robot is ready but holding. (Never step on the "HOLD" foot switch before clicking the "START" button.)

3. **Step on the "HOLD" foot switch to enable the robot.**

4. **If you want to stop the robot or when the insertion is complete, release the "HOLD" foot switch first and then click the "STOP" button on the Action Panel.**

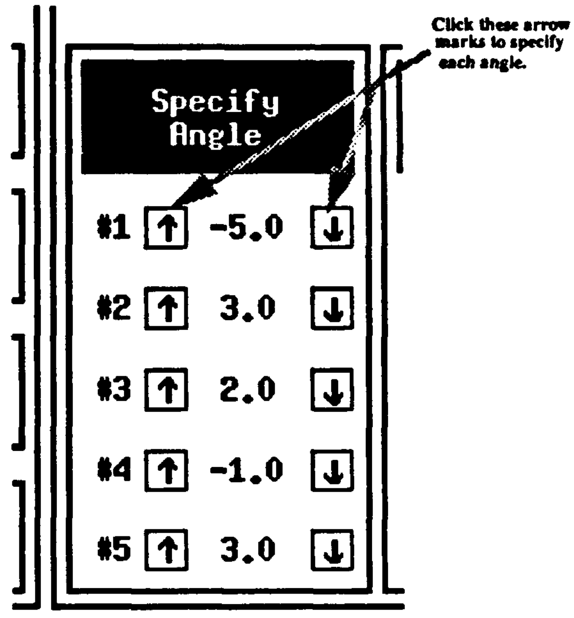
2.3 Teleoperation

Currently, TORCS supports four different methods of teleoperation: Angle Specification, Gestural Control, Joystick Control, and the Hand Controller. The selection of these devices are done through the Screen Interface. Some of devices have advanced options. This section contains instructions for using each of these devices.

2.3.1 Angle Specification

This feature allows an operator to directly manipulate the joint angles of the robot using the mouse. Angles are specified relative to the current joint angles and are, thus, just angle changes. This feature is used as follows:

Figure 2.7 "Specify Angle" for Teleoperation



1. **Click the "Specify Angle" button on the Parameter Panel.**

2. **Specify the angle change for each joint.**
   You can specify the angle change for each joint by clicking the mouse (or the track ball) button on the up or down arrow marks. The numbers on the panel indicate degrees of change. Each click of the mouse yields a change of 1/2 degree.

3. **When ready, click the "START" button.**
   Don't step on the "HOLD" foot switch before clicking the "START" button. The "HOLD" indicator will now light to show that the robot is ready.

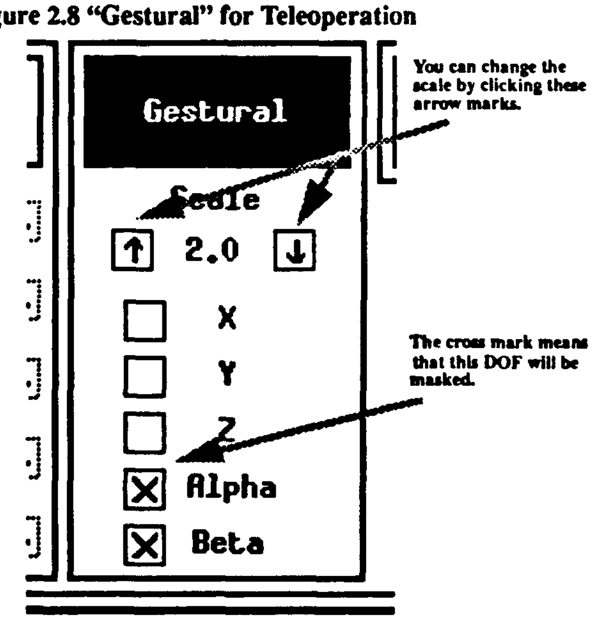
4. **Step on the "HOLD" foot switch.**

5. **When finished, release the "HOLD" foot switch first and then click the "STOP" button on the Action Panel.**

2.3.2 Gestural Control

A hand-held six degree of freedom Polhemus can be used for Gestural Control. Five degrees of freedom are used to control the tip of the robot in Cartesian space. TORCS can set several options to the gestural control system including the scale of movement, and the active degrees of freedom. When a degree of freedom is deactivated, it is taken over by the Real-Time Controller.

Figure 2.8 "Gestural" for Teleoperation



1. **Click the "Gestural" button on the Parameter Panel.**

2. **Specify the scale.**
   he default scale is one to one but this can be changed by clicking either the up or the down arrow mark.

3. **Select masked Degrees of Freedom.**
   Click the appropriate boxes to select any degrees of freedom which you would like to be controlled by the real-time Controller. Selected boxes will be marked with an X.

2. **When ready, click the "START" button on the Action Panel to start the insertion.**
   The "HOLD" light should be on, indicating that the robot is ready but holding. (Never step on the "HOLD" foot switch before clicking the "START" button.)

3. **Step on the "HOLD" foot switch to enable the robot.**

4. **If you want to stop the robot or when the insertion is complete, release the "HOLD" foot switch first and then click the "STOP" button on the Action Panel.**

2.3.3 The Joystick

An operator can also select the joystick as a device for teleoperation. This device has no special options and is used as follows:

1. Click the "Joystick" button on the Parameter Panel.

2. When ready, click the "START" button on the Action Panel to start the movement.
The "HOLD" light should be on, indicating that the robot is ready but holding. (Never step on the "HOLD" foot switch before clicking the "START" button.)

3. Step on the "HOLD" foot switch to enable the robot.
Use the joystick to control the movement of the robot.

4. If you want to stop the robot or when the movement is complete, release the "HOLD" foot switch first and then click the "STOP" button on the Action Panel.

2.3.4 The Hand Controller

The Hand Controller is just like a miniature robot of the $(SM)^2$. It has the same configuration and number of degrees of freedom as the real robot. Currently, there are no special options for this device which is used as follows:

1. Click the "Hand Controller" button on the Parameter Panel.

2. When ready, click the "START" button on the Action Panel to start the movement.
The "HOLD" light should be on, indicating that the robot is ready but holding. (Never step on the "HOLD" foot switch before clicking the "START" button.)

3. Step on the "HOLD" foot switch to enable the robot.
Use the hand controller to control the movement of the robot.

4. If you want to stop the robot or when the movement is complete, release the "HOLD" foot switch first and then click the "STOP" button on the Action Panel.

2.4 Rescue Procedures

TORCS is designed to be as robust as possible in the face of a variety of malfunctions or other unexpected events. In case of problems, it should be fairly easy for the operator to rescue the robot and to continue normal operation. Described below are several problem conditions with appropriate recovery methods.

<u>**Rescue Procedure (case: The initial operation mode is Autonomous Stepping)**</u>

<u>*situation1: During walking, the $(SM)^2$ fails to insert sucessfully.*</u>

rescue1: Release the "HOLD" foot switch.
When you release the "HOLD" foot switch, the "HOLD" light will turn on. (The servo for each motor will continue to operate, holding the tip in the current position.)

rescue2: Click one of device buttons for the Teleoperation.
Choose one of the four devices which you will use to rescue the robot.

rescue3: Click the "START" button on the Action Panel, if you are ready.

rescue4: Step on the "HOLD" foot switch.
This activates the chosen device.

rescue5: Complete the insertion via the teleoperation device.

situation2: The $(SM)^2$ still cannot insert the tip into that node, because of a defect in the node.

rescue1: Release the "HOLD" foot switch.

rescue2: If the previous Rescue Procedure is still going on, click the "STOP" button on the Action Panel.

rescue3: Click the "COARSE" button and set the parameters for the Coarse Motion.
Select a new target node to detour around the defective node.

rescue4: Click the "START" button on the Action Panel.

rescue5: Step on the "HOLD" foot switch, the Coarse Motion will be started.

rescue6: Release the "HOLD" foot switch when the tip is close to the new target node.

rescue7: Click the "STOP" button on the Action Panel.

rescue8: Complete the insertion by using the Fine Motion and Insertion phases.
After the insertion is completed, Autonomous Motion may be used to continue the original walk.

situation3: After making a detour, you want the $(SM)^2$ to return to the original path.

rescue1: Click the "START" button on the Action Panel again.
If the robot is on a waypoint of the original path, it will continue on the original path. Otherwise, an error message will be displayed and the operator must specify a new target.

rescue2: If the $(SM)^2$ will continue the original path, step on the "HOLD" foot switch.

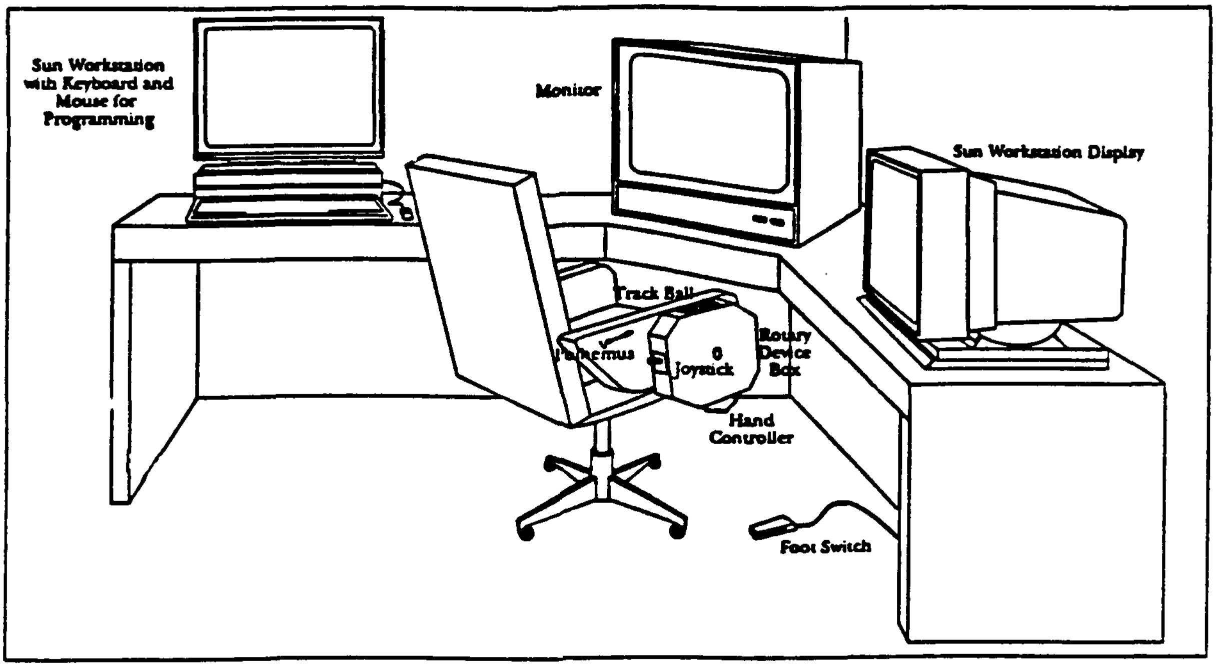
rescue3: When the walk is complete, release the "HOLD" foot switch first and then click the "STOP" button on the Action Panel.

situation4: After making a detour, you want to cancel the original operation.

rescue1: Click the "STOP" button on the Action Panel again.
The operation is terminated.

Appendix A

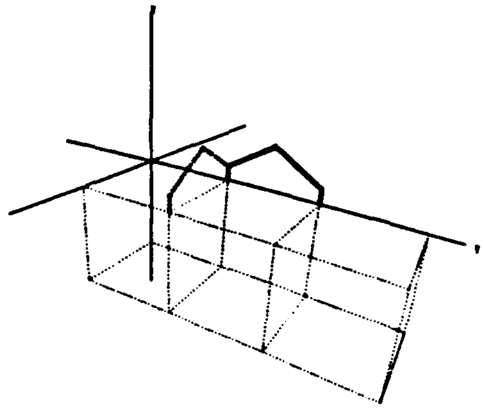
Control Station for $(SM)^2$



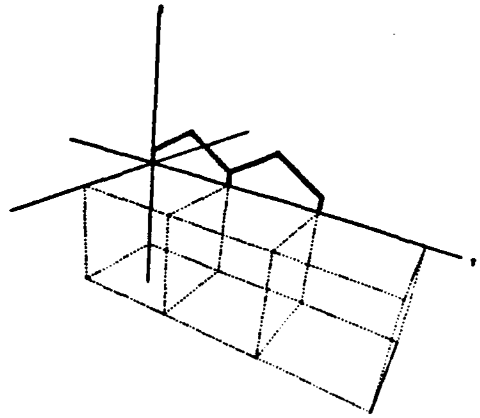
Appendix B

Step Catalog

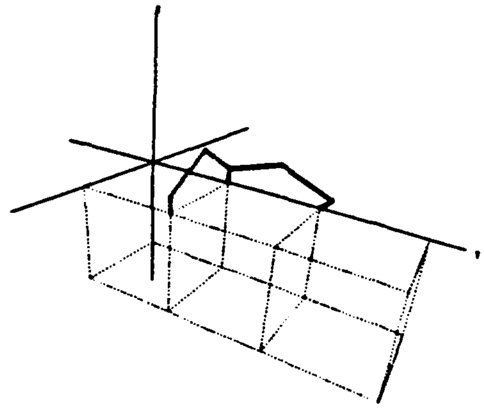
(1) Simple 90 [deg] Step



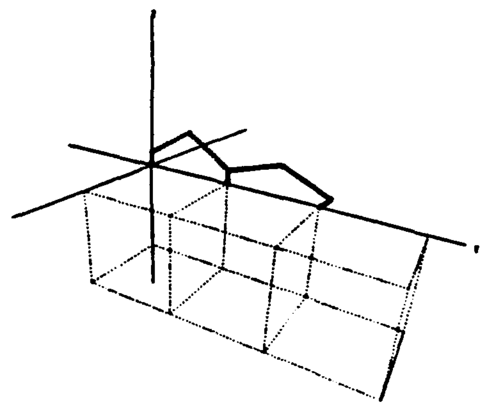
(2) Simple 180 [deg] Step



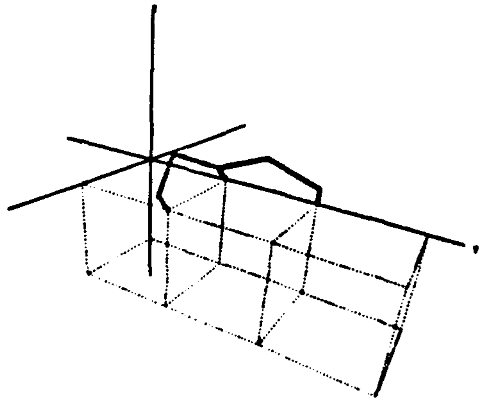
(3) 90 [deg] Step in the Plane from 45 [deg] Tip Hole



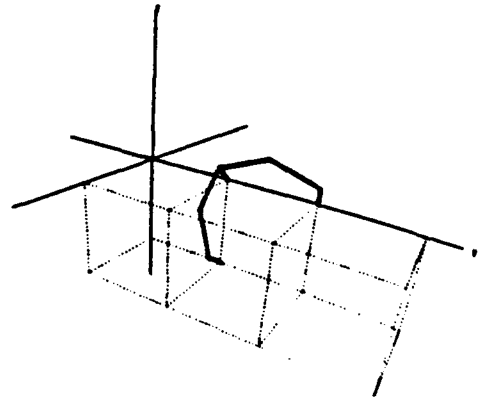
(4) 180 [deg] Step in the Plane from 45 [deg] Tip Hole



(5) 90 [deg] Step to 45 [deg] Plane



(6) 180 [deg] Step to 90 [deg] Plane



Appendix C

Structure of Strut Data Base

Strut	NodeA*1.	NodeB	*3. Node A						Node B						Strut *4.
No.	No.	No.	+X	-X	+Y	-Y	+Z	-Z	+X	-X	+Y	-Y	+Z	-Z	Dir.
0	-1 *2.	0	7		1			21	6					20	2
1	2	-1	8		2			22	7			0		21	2
2	-3	2	9					23	8			1		22	2
3	4	-5		9				24		8		4		25	2
4	-5	6		8	3			25		7		5		26	2
5	6	-7		7	4			26		6				27	2
6	-7	0			5			27			0			20	1
7	6	-1			4	5		26			1	0		21	1
8	-5	2			3	4		25			2	1		22	1
9	4	-3				3		24				2		23	1
10	9	-8	17		11		21		16				20		2
11	-10	9	18		12		22		17			10	21		2
12	11	-10	19				23		18			11	22		2
13	-12	13		19			24			18		14	25		2
14	13	-14		18	13		25			17		15	26		2
15	-14	15		17	14		26			16			27		2
16	15	-8			15		27				10		20		1
17	-14	9			14	15	26				11	10	21		1
18	13	-10			13	14	25				12	11	22		1
19	-12	11				13	24					12	23		1
20	0	-8	6		0				16		10				3
21	-1	9	7		1	0			17		11	10			3
22	2	-10	8		2	1			18		12	11			3
23	-3	11	9			2			19			12			3
24	4	-12		9		3				19		13			3
25	-5	13		8	3	4				18	13	14			3
26	6	-14		7	4	5				17	14	15			3
27	-7	15		6	5					16	15				3

*1. Node A is located on the positive end of the strut and Node B is located on the negative end of the strut.

*2. These numbers are the node numbers and negative values mean that only foot A can be inserted into these nodes (because currently the $(SM)^2$ cannot insert both ends into the same node).

*3. These numbers are the Strut Numbers.

*4. These numbers correspond to the direction with respect to the world (truss) coordinate as follows: 1: X direction, 2: Y direction, 3: Z direction

Appendix D

Setting Up the Teleoperated Robot Control System

This appendix covers the setup of the TORCS system. The tutorial assumes that the operator has standard knowledge of the UNIX operating system and the X window environment. Typically, the $(SM)^2$ uses three Sun workstations (See Fig. 1.1 Hardware Configuration.):

a) one Sun 3 workstation running Sun OS 4.1.1 supporting:
  at least one MC68020 Real-Time Processing Unit (RTPU)
  A to D, D to A and other communications boards
b) one Sun 4 used for the Screen Interface
c) one Sun 4 used for the Vision System supporting:
  one to three Matrox frame buffer boards

In order to use the TORCS, you must ensure that the Sun 3 workstation you are using has proper hardware, and that the CHIMERA II software is installed. (See *CHIMERAII Real-Time Programing Environment* to set up CHIMERAII environment.) The TORCS user interface runs only under the X11 window system.

A) Setting Up Your Environment Variables

Before running any of the software, it is good idea to set up the following environment variables. It is suggested that the appropriate **setenv** commands be placed in your **.login** file.

SM2_DIR: This variable should be set the base directory of the $(SM)^2$ software. Normally it should be **/usr/sm2**. Check with your local system administrator for the proper path on your system.

PATH: Include the path **$(SM2_DIR)/bin** in your path variable.

B) Executable Files for TORCS

The following are the executable files that comprise the TORCS system. These are located in $(SM2-DIR)/bin. The order of execution of the various programs is not critical.

- **Flex**: This is the executable file for the Real-Time Controller.
- **glbdaemon**: This is the executable file for the User Integration daemon.
- **smvision**: This is the executable file for the real-time vision system.
- **sm2**: This is the executable file for the Screen Interface.

Note: The executable file for the Robot Simulator is not included listed here.

C) Startup Procedure

1. Run the vision system program, first.

Type the following command in one window which is connected to the Sun 4 workstation for the vision system.

% smvision

The following message will appear

*** Opening /usr/pomerlea/sm2/aux_files/sm2_param_file.txt to read user parameters
Are you ready to digitize? [yes]

When ready, hit the RETURN key.

2. Enter the CHIMERA II Environment.
To execute the Teleoperated Robot Control System program, it is necessary to enter the CHIMERA II environment. In one of the Sun 3 workstation windows (you can also enter the X window system environment on a Sun 3 workstation. But if you can remote login to the Sun 3 from the X window environment on a Sun 4, it is good idea you operate from one display) type the following command.

% chim

For the detailed information on CHIMERA II, see the manual *CHIMERA II Real-Time Programing Environment Version 1.02*.

3. Execute the Real-Time Control program.
Under CHIMERA II environment, execute the Real-Time Controller.

CHIM:*rtpuname*> ex Flex

The following menu will appear in the window.

Operation ***************
1: Set task parameters & Trajectories
2: Set control parameters
3: Calibrate sensors
4: Run Robot
5: Send data
6: Option..
7: Demonstrate Walking
8: Demonstrate Walking with UI
0: End (0 to 8) [3]

4. Choose 8 to start the Real-Time Controller.

5. Execute the "daemon" process which handles User Integration.
In another window which is connected with the Sun 3 workstation, type the following command.

% glbdaemon

6. Execute the Screen Interface.
Type the following command in one window which is connected to the Sun 4 window for the Screen Interface.

% sm2

You are ready to use the Teleoperated Robot Control System.

Appendix E

Source Code

A) Code for the Real-Time Controller

Source code for the Teleoperated Robot Control System can be found in **$(SM2_DIR)/rtc/src**. All needed header files are in **$(SM2_DIR)/rtc/include.**

Some source and header files which are used for communication are in **$(SM2_DIR)/gds/host** and **$(SM2_DIR)/gds/include.**

B) Code for the "daemon" Data Server

Source code is in **$(SM2_DIR)/gds/daemon** and header files are in **$(SM2_DIR)/gds/include.**

C) Code for the Vision System

Source code is in **$(SM2_DIR)/vision/src** and header files are in **$(SM2_DIR)/vision/include.**

D) Code for the Screen Interface

Source code is in **$(SM2_DIR)/si/src** and header files are in **$(SM2_DIR)/si/include.** Some header files which are needed for communication are in **$(SM2_DIR)/gds/include.**

References

[1] H. B. Brown, Jr., M. Friedman, and T. Kanade, "Development of a 5-DOF walking robot for space station application", *Proceedings of IEEE International System Engineering Conference*, 1990, Pittsburgh.

[2] H. Ueno, Y. Xu, H. B. Brown, Jr., M. Ueno, and T. Kanade, "On control and planning of a space station robot walker", *Proceedings of IEEE International System Engineering Conference*, 1990, Pittsburgh.

[3] D. B. Stewart, D. E. Schmitz, and P. K. Khosla, "CHIMERA II Real-Time Programming Environment version 1.02", *Advanced Manipulator Laboratory the Robotics Institute and Department of Electrical and Computer Engineering Carnegie Mellon University technical report*, 1990.